FLIESLER MEYER LLP

INTELLECTUAL PROPERTY LAW

650 CALIFORNIA STREET • FOURTEENTH FLOOR
SAN FRANCISCO • CALIFORNIA 94108
TELEPHONE 415.362.3800
FACSIMILE 415.362.2928
INTERNET WWW.FDML.COM

TO: Examiner Thuy Chan

FAX NO.: 571-273-8570

FROM: Kuiran (Ted) Liu

RE: **UNOFFICIAL DRAFT**
Inventor(s): Josh Eckels, *et al.*
Appln. No.: 10/784,346
Confirm. No.: 2353
Filed: February 23, 2004
Title: SYSTEMS AND METHODS FOR MULTI-VIEW DEBUGGING ENVIRONMENT

DATE: July 15, 2008 Total Pages : 13

Original will follow by mail: No

If you do not receive all of the pages, please call Narissa M. Besada at 415.362.3800.

MESSAGE (if any):

This facsimile is intended only for the addressee and those authorized by the addressee to receive it. Any use, dissemination, distribution or copying of this facsimile by any others is prohibited. Any others receiving this facsimile are requested to notify FLIESLER MEYER LLP immediately by telephone or fax and to return the original facsimile to FLIESLER MEYER LLP.

UNOFFICIAL DRAFT – NOT TO BE ENTERED

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application	PATENT APPLICATION
Inventor(s): Josh Eckels, *et al.*	Art Unit: 2192
Appln. No.: 10/784,346	Examiner: Dao, Thuy Chan
Confirm. No.: 2353	
Filed: February 23, 2004	
Title: SYSTEMS AND METHODS FOR MULTI-VIEW DEBUGGING ENVIRONMENT	**Customer No. 23910**

REPLY TO OFFICE ACTION UNDER 37 C.F.R. 1.111

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

In response to the Office Action of May 12, 2008, please amend the above-identified application as followed:

Amendments to the Claims begins on page 2 of this paper.

Remarks/Arguments begin on page 10 of this paper.

AMENDMENTS TO THE CLAIMS

Please amend claims 1, 7, 11, 12, 13, 18, 20, 21, 23, 28, 30, 31, 32 and 33 as shown below. All pending claims are reproduced below, including those that remain unchanged.

1. (Currently Amended) A computer-enabled system embodied in a storage medium to provide a software debugging environment, comprising:

   an executing software program containing at least one data structure;

   at least one abstract view capable of displaying and/or editing at least one abstract content of the at least one data structure, wherein the abstract content of the at least one data structure constitutes attributes of interest during the execution of the executing software program rather than underlying physical data structures used to represent the abstract content; and

   at least one filter capable of extracting and formatting the contents of interest from the underlying physical data structures and defining a displaying and/or editing property of the at least one abstract view, such property can include at least one of: which of the at least one abstract content is displayed, a format in which it is displayed, and how it is edited.

2. (Original) The system according to claim 1, wherein:

   the system is at least partially implemented using Java language.

3. (Original) The system according to claim 1, further comprising:

   at least one editor associated with the at least one abstract view capable of at least one of:

   allowing the at least one abstract content to be modified through the at least one abstract view; and

   validating an input value to the at least one abstract content against an allowed value for the at least one abstract content.

4. (Original) The system according to claim 1, wherein:

the at least one abstract view is capable of presenting the at least one abstract content of the at least one data structure without showing a physical implementation of the at least one data structure.

5. (Original) The system according to claim 1, wherein:
each of the at least one abstract view can be individually selected for display.

6. (Original) The system according to claim 1, wherein:
Two or more of the at least one abstract view are capable of displaying and/or editing the same one of the at least one abstract content without being deadlocked.

7. (Currently Amended) The system according to claim 1, wherein:
the at least one filter can be defined via configuration information stored in a file, which can be ~~an~~ ~~XML~~ file in a markup language.

8. (Original) The system according to claim 1, further comprising:
a component capable of interactively performing at least one of:
selecting a subset of the at least one of abstract view for display; and
defining the displaying and/or editing property of the at least one filter.

9. (Original) The system according to claim 8, wherein:
the component can be realized via an interface to an Integrated Development Environment (IDE).

10. (Original) The system according to claim 1, further comprising:
at least one component capable of supporting the debugging of a JSP page and a machine generated servlet that implements the JSP page.

11. (Currently Amended) The system according to claim 10, wherein:

the at least one component can perform at least one of:

extracting and displaying a code and/or a content of interest, and mapping them to a format used in a source code in a ~~JSP~~ server page, for use with executing a ~~JSP~~ servlet;

following an execution path through at least one level of redirection using at least one tag;

extracting and manipulating a streaming data from a content of a buffer used to transmit and receive the streaming data; and

setting at least one break point in a ~~JSP~~ server page and stepping through the execution of the page based on the displaying property.

12. (Currently Amended) The system according to claim 11, wherein:

the streaming data can be extracted by inserting a wrapper or "writer" class around the ~~JSP~~ servlet.

13. (Currently Amended) A method to provide a software debugging environment, comprising:

displaying and/or editing at least one abstract content of at least one data structure in an executing software program via at least one abstract view, wherein the abstract content of the at least one data structure constitutes attributes of interest during the execution of the executing software program rather than underlying physical data structures used to represent the abstract content; and

extracting and formatting the contents of interest from the underlying physical data structures and defining a displaying and/or editing property of the at least one abstract view via at least one filter, such property can include at least one of: which of the at least one abstract content is displayed, a format in which it is displayed, and how it is edited.

14. (Original) The method according to claim 13, further comprising:

allowing the at least one abstract content to be modified through the at least one abstract view; and validating an input value to the at least one content against an allowed value for the at least one content.

15. (Original) The method according to claim 13, further comprising:

presenting the at least one abstract content of the at least one data structure without showing a physical implementation of the at least one data structure.

16. (Original) The method according to claim 13, further comprising:

selecting each of the at least one abstract view individually for display.

17. (Original) The method according to claim 13, further comprising:

displaying and/or editing the same one of the at least one abstract content via two or more of the at least one abstract view without being deadlocked.

18. (Currently Amended) The method according to claim 13, further comprising:

defining the at least one filter via configuration information stored in a file, which can be a~~n~~ ~~XML~~ file in a markup language.

19. (Original) The method according to claim 13, further comprising:

interactively performing at least one of:
selecting a subset of the at least one of abstract view for display; and
defining the displaying and/or editing property of the at least one filter.

20. (Currently Amended) The method according to claim 13, further comprising:

supporting the debugging of a ~~JSP~~ server page and a machine generated servlet that implements the ~~JSP~~ server page.

21. (Currently Amended) The method according to claim 20, further comprising:

extracting and displaying a code and/or a content of interest, and mapping them to a format used in a source code in a ~~JSP~~ server page, for use with executing a ~~JSP~~ servlet;

following an execution path through at least one level of redirection using at least one tag;

extracting and manipulating a streaming data from a content of a buffer used to transmit and receive the streaming data; and

setting at least one break point in a ~~JSP~~ server page and stepping through the execution of the page based on the displaying property.

22. (Original) The method according to claim 21, wherein:

the streaming data can be extracted by inserting a wrapper or "writer" class around the JSP servlet.

23. (Currently Amended) A machine readable medium having instructions stored thereon that when executed by a processor cause a system to:

display and/or edit at least one abstract content of at least one data structure in an executing software program via at least one abstract view, wherein the abstract content of the at least one data structure constitutes attributes of interest during the execution of the executing software program rather than underlying physical data structures used to represent the abstract content; and

extract and format the contents of interest from the underlying physical data structures and define a displaying and/or editing property of the at least one abstract view via at least one filter, such property can include at least one of: which of the at least one abstract content is displayed, a format in which it is displayed, and how it is edited.

24. (Original) The machine readable medium of claim 23, further comprising instructions that when executed cause the system to:

allow the at least one abstract content to be modified through the at least one abstract view; and validate an input value to the at least one abstract content against an allowed value for the at least one content.

25. (Original) The machine readable medium of claim 23, further comprising instructions that when executed cause the system to:

present the at least one abstract content of the at least one data structure without showing an physical implementation of the at least one data structure.

26. (Original) The machine readable medium of claim 23, further comprising instructions that when executed cause the system to:

select each of the at least one abstract view individually for display.

27. (Original) The machine readable medium of claim 23, further comprising instructions that when executed cause the system to:

display and/or edit the same one of the at least one abstract content via two or more of the at least one abstract view without being deadlocked.

28. (Currently Amended) The machine readable medium of claim 23, further comprising instructions that when executed cause the system to:

define the at least one filter via configuration information stored in a file, which can be a~~n~~ ~~XML~~ file in a markup language.

29. (Original) The machine readable medium of claim 23, further comprising instructions that when executed cause the system to:

interactively perform at least one of:

selecting a subset of the at least one of abstract view for display; and

defining the displaying and/or editing property of the at least one filter.

30. (Currently Amended) The machine readable medium of claim 23 further comprising instructions that when executed cause the system to:

support the debugging of a ~~JSP~~ server page and a machine generated servlet that implements the ~~JSP~~ server page.

31. (Currently Amended) The machine readable medium of claim 30, further comprising instructions that when executed cause the system to:

extract and display a code and/or a content of interest, and mapping them to a format used in a source code in a ~~JSP~~ server page, for use with executing a ~~JSP~~ servlet;

follow an execution path through at least one level of redirection using at least one tag;

extract and manipulate a streaming data from a content of a buffer used to transmit and receive the streaming data; and

set at least one break point in a ~~JSP~~ server page and step through the execution of the page based on the displaying property.

32. (Currently Amended) The machine readable medium of claim 31, wherein:

the streaming data can be extracted by inserting a wrapper or "writer" class around the ~~JSP~~ servlet.

33. (Currently Amended) A computer-enabled system embodied in a storage medium to provide a software debugging environment, comprising:

means for displaying and/or editing at least one abstract content of at least one data structure in an executing software program via at least one abstract view, wherein the abstract content of the at least one data structure constitutes attributes of interest during the execution of the executing software program rather than underlying physical data structures used to represent the abstract content; and

means for extracting and formatting the contents of interest from the underlying physical data structures and defining a displaying and/or editing property of the at least one abstract view via at least one filter, such property can include at least one of: which of the at least one abstract content is displayed, a format in which it is displayed, and how it is edited.

34. (Canceled).

REMARKS/ARGUMENTS

The above Amendment and these Remarks are in response to the Office Action mailed May 12, 2008. Claims 1-33 were pending prior to the outstanding Office Action. In the Office Action, the Examiner rejected claims 1-33. This Response amends claims 1, 7, 11, 12, 13, 18, 20, 21, 23, 28, 30, 31, and 33, leaving for the Examiner's consideration claims 1-33. Reconsideration of the rejections is respectfully requested.

CLAIM REJECTIONS

Claims 7 and 11 are objected to because of minor informalities.

Applicant respectfully submits that the claims as amended now conform to the requirements of the Examiner.

CLAIM REJECTIONS – 35 USC § 102 and § 103

Claims 1-6, 8-9, 13-17, 19, 23-27, 29, and 33 are rejected under 35 USC 102(e) as being anticipated by U.S. Patent Publication No. 2005/0278585 to Spencer.

Claims 7, 10-12, 18, 20-22, 28 and 30-32 are rejected under 35 USC 103(a) as being unpatentable over Spencer in view of Charisius (art of record, U.S. Patent No. 7,051,316.

Here, independent claims 1, 13, 23, and 33 are all amended to include that "*a filter capable of extracting and formatting the contents of interest from the underlying physical data structures and defining a displaying and/or editing property of the at least one abstract view*," in addition to state that "*wherein the abstract content of the at least one data structure constitutes attributes of interest during the execution of the executing software program rather than underlying physical data structures used to represent the abstract content.*"

The above amendment can be explained by the example described in **[0005]**. In the example, a developer can use a data structure called a List to represent an ordered collection of items on an invoice. In the present invention embodied in claim 1, the debugger can allow the developer to see the abstract content of the List, such as the list of items and their attributes of interest during the execution of the executing software program (e.g., quantity, price, description), instead of the physical structure such as a bunch of pointers that are used to implement the List data structure using a linked list of nodes.

Different from the present invention, Spencer focuses on showing the expressions and information in a floating or movable window, either above the source code, or attached to the source code in the locations that they are relevant to. (Paragraph **0022**, Line **10-13**). Applicant respectfully submits that, in Fig. 4, Local window 320 shows an integer instance "i=0," which is the underlying physical data structure. Similarly, auto window 310 shows a "ListViewItem" which the most relevant variables for debugging at that point in the code. (Paragraph **0043**, Line **4-5**). However, there is no indication in Spencer shows that "ListViewItem" is not the underlying physical data structure. Therefore, there is no need of extracting and formatting the contents of interest from the underlying physical data structures in Spencer. Hence, Spencer cannot anticipate the present invention or render the present invention obvious, since only variables and expressions in the physical structures (the linked list in the above example), not the abstract content (the List in the above example), are shown in Spencer. Therefore, independent claims 1, 13, 23, and 33 should all be in allowable condition.

In addition, dependent claims 2-12, which are based on independent claim 1; dependent claims 14-22, which are based on independent claim 13; and dependent claims 24-32, which are based on independent claim 23, should also be in allowable condition.

Conclusion

In light of the above, it is respectfully submitted that all of the claims now pending in the subject patent application are allowable, and Applicants respectfully request that a timely Notice of Allowance be issued in this case.

The Commissioner is authorized to charge any underpayment or credit any overpayment to Deposit Account No. 06-1325 for any matter in connection with this response, including any fee for extension of time, which may be required.

Respectfully submitted,

Date: July 15, 2008

By: /Kuiran (Ted) Liu/
Kuiran (Ted) Liu
Reg. No. 60,039

FLIESLER MEYER LLP
650 California Street, Fourteenth Floor
San Francisco, California 94108
Telephone: (415) 362-3800